uc3m | Universidad Carlos III de Madrid

University Degree in Computer Science and Engineering
and Business Administration
2018-2019

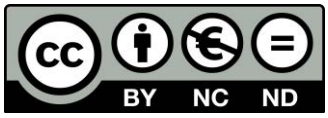
Bachelor Thesis

"Estimation of Depth Maps from Monocular Images using Deep Neural Networks"

Daniel Sarmiento Rocha

Miguel Ángel Patricio Guisado

Colmenarejo, 17 June 2019



Abstract

Computer vision tasks have seen recent improvements thanks to the development of deep learning and high-end hardware. One of these tasks is depth perception, which involves extracting three-dimensional information from two-dimensional elements like images and constructing a depth map. This kind of information is useful in many domains such as autonomous vehicles or scene reconstruction for augmented and virtual reality. Humans and some other animals achieve this by using binocular vision (vision from two images) and some algorithms have been developed to imitate this process. However, recent progress has enabled the advancement of other approaches that allow monocular vision algorithms to accomplish decent depth maps. In this thesis two monocular deep learning methods (Monodepth and DenseDepth) are explored and compared to each other (and with binocular and monocular approaches in general). This experiment is conducted by exposing the two methods to images that have not been seen during training and performing a qualitative analysis of their results in two different scenarios: indoors and outdoors. Both Monodepth and DenseDepth are able to produce depth maps, but DenseDepth results are more promising and reliable. Results show the importance of the training domain, as the accuracy is affected by the choice of pre-trained models, as well as the collection and selection of data. It is still an open problem and seems unlikely that monocular depth perception could replace other sensors in critical systems like autonomous driving. However, it could still be a great complement or useful in other products or domains like photography.

Key words: computer vision, depth perception, monocular vision, stereo matching, deep learning

Contents

Figures and Tables

Chapter 1

Introduction

1.1 Introduction to the task

Humans and many other animals have no problem understanding three dimensional structures from images thanks to our binocular vision. This process is called **depth perception**[1] both in biological and computer vision. Nonetheless, it has been and remains one of the most challenging issues and unsolved problems in this field of computer science. Computer vision can be defined both as the "scientific field that extracts information out of digital images" and "building algorithms that can understand the content of images and use it for other applications" [1]. The applications of computer vision are broad and diverse: special effects, 3D urban modelling, scene recognition, face detection, self-driving cars, automatic checkout, vision-based interaction, augmented reality and virtual reality just to name a few of them [1].

In computer vision, depth perception consists on estimating distances from data provided by sensors [2], usually cameras, that mimics the behavior of human eyes. This kind of information is very valuable for the development of robust guidance systems in autonomous vehicles. If accurate enough, estimating depth from images can prevent them from having to rely on multiple kinds of sensors such as radar and ultrasonic sensors. This would not only be more affordable, but also less power intensive and bulkier, which could be problematic when working with small unmanned vehicles or other small devices. Apart from robot navigation, this is also an important issue in virtual and augmented

[1] Also: depth extraction, depth estimation or depth inference

reality, depth measurements, environment reconstruction and other aspects of production, security, defense, exploration and entertainment [3].

Using two cameras and a stereo matching algorithm we can construct a depth map, an image whose pixel values are directly proportional to the distance from the lenses who captured the image. However, this still depends on redundant hardware, two cameras, and an array of different conditions (rectification, calibration, etc.) that are necessary for the stereo matching algorithm to perform accurately. With the help of deep learning, a single image from a single monocular camera is enough to achieve decent depth perception.

This work is an exploration of the architecture, performance and results of the deep learning algorithms that can produce depth maps from a single image. It is also discussed whether it is a valid approach as an alternative to binocular stereo matching, its limitations and its role in the development of cost-effective depth aware systems for autonomous navigation.

1.2 Goals

This Bachelor's Thesis aims to obtain depth maps from a single input image that can be useful for the tasks that benefit from the 3D information of a scene. Thus, the following objectives have been established:

- Briefly present **how stereo matching algorithms work** and evaluate their performance.
- Analyze **how monocular and binocular depth perception compare** to each other.
- Explore and **compare two monocular depth perception algorithms** and its architecture.
- Adapt and reproduce the selected algorithms in the available environment, **evaluate the results** (qualitatively) **and its suitability** for different relevant tasks.
    - Produce **depth maps from images** that the algorithms have **never been trained or test on**.

- **Propose future lines of research** related to monocular depth perception based on the results of this thesis.

1.3 Methodology

The first part of this thesis is mostly a **literary review** of recent and relevant research in the field of computer vision, specifically the task of depth estimation from both stereo and single images. Above all, emphasis was made on studies and research paper that treated this problem with machine or deep learning. This study was conducted relying mainly on academic research databases such as Google Scholar, IEEE Xplore Digital Library and the Institutional Repository of Universidad Carlos III de Madrid.

The second part consists in an **experiment** regarding the reproduction of the selected deep learning methods and the qualitative analysis of their results, in this case depth/disparity maps. The details about how the experiments were conducted, the environment used, data fed to the models and the models themselves are discussed in Chapter 4.

1.4 Structure

For the purpose of achieving the objectives mentioned before, the main body of this document is divided into four chapters.

The first chapter after this, Chapter 2, briefly introduces what binocular vision is and how it works. It is followed by an overview on how binocular vision is simulated in the computer vision field via stereo matching algorithms. It ends with a brief analysis about the performance and efficiency of the most relevant methods and a demonstration of the results that it was possible to reproduce.

Chapter 3 introduces the concept of monocular vision and how it translates to the task of depth estimation in computer vision. Two methods are then selected to illustrate different approaches for the task of estimating depth from a single RGB image. The study of these

two models in this chapter is focused on how they work and how well they perform compared to other state-of-the-art techniques.

The fourth chapter deals with the degree of reproducibility of the methods introduced in the previous chapter and a first approach to the solution. It starts with an explanation showing how the experiments were conducted and a description of the environment as well as which pre-trained models were used and why. Results are then presented and compared in two different domains to help best show which method displays better qualitative results.

The last chapter shows the main conclusions of this Bachelor's Thesis and discusses future related work that was not included in this thesis due to time constraints and/or because it exceeded the main focus of the study.

1.5 State of the art

The amount of research problems that are benefitting from the use of machine and deep learning approaches is growing in many different fields (cancer diagnosis, autonomous driving, spam detection, etc.). Computer vision is one of the areas that have recently experienced a vast amount of improvement with the advances in neural networks, deep learning and hardware costs. Depth estimation, the inference of a dense depth map from an input image(s), is one of the tasks in this area that has seen some improvements in both performance and accuracy over the last years.

When working with images, traditional neural networks like the MLP are not widely used anymore in favor of Convolutional Neural Networks or CNNs/ConvNets. These types of neural networks share a lot with traditional (fully-connected) neural networks: they are made up of neurons with weights and biases that adapt to the task and they have a loss function that help adjust those values (learning).

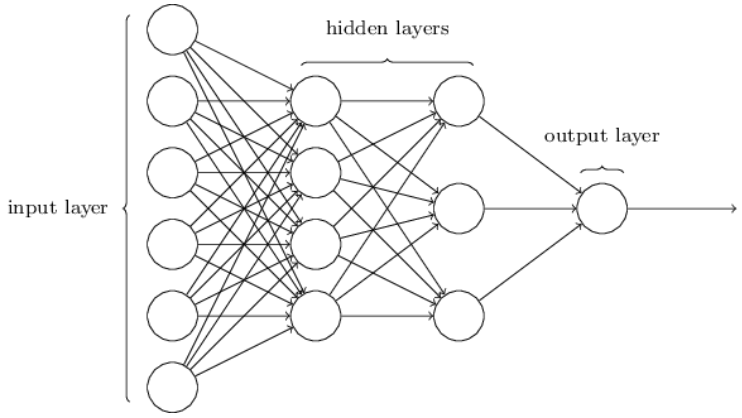


Figure 1. Example of a traditional neural network (MLP) [4].

However, CNNs treat always the input as an image (does not have to be an image), which makes it the best choice for computer vision tasks. CNNs scale well to full images, as opposed to traditional neural networks. This is best illustrated with an example. For an image of size $32 \times 32 \times 3$ (32 pixels wide, 32 pixels high and 3 color channels for red, green and blue), the first neuron of the first hidden layer would have 3072 weights. But a square of 32 pixels is not likely to be our input image in complex problems. Suppose a more natural image size of $640 \times 480 \times 3$ and the first neuron of the first hidden layer would have 921600 weights. Almost a million weights in just one neuron in the first hidden layer translates to a very inefficient neural network in the end. On the other hand, CNNs have three-dimensional neural structures called activation volumes. Instead of fully-connected hidden layers, the most important layers they have are convolutional layers, which are responsible for detecting features from the image such as edges, patterns, shapes and textures. This is done with relatively small matrices of values called filters or kernels which contain the weights. These matrices convolve or slide across the image and produce another activation volume. For the previous example, we could have only 75 ($5 \times 5 \times 3$) weights for the $640 \times 480 \times 3$ image, much more manageable than 921600 [5, 6]. Additionally, pooling layers (average or maximum pooling) help reduce the size of the image, which in turn helps with computation. This can be best observed in the following figure.

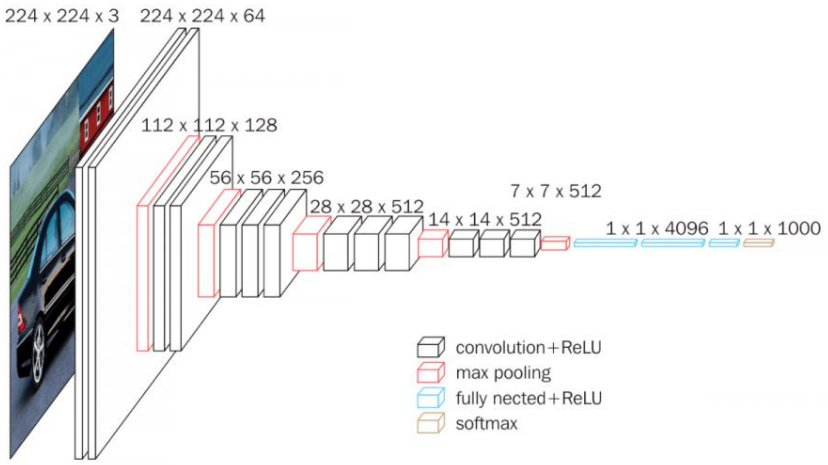


Figure 2. Example of a CNN for image recognition [7]

Even though many of the modern solutions for this task use Convolutional Neural Networks, they differ widely from one another and can be divided into four subgroups of learning based approaches: monocular, multi-view, transfer learning and encoder-decoder, which are non-exclusive.

The approaches that use only one RGB image as input are called **monocular depth estimation algorithms**. Eigen et al. [8] developed a method using two connected deep neural networks with two individual goals: the first one outputs a coarse depth map based on the entire image and the second network combines this output with the original RGB image as input to refine the initial rough depth map. Its results are not considered impressive nowadays, but it set the path for others to follow and improve. A much more recent approach by Hao et al. [9] proposes another network architecture consisting on two differentiated phases: the first one uses technology inspired by semantic segmentation networks to extract features from the image and the second on merges these features using attention mechanism to generate a depth map. The authors claimed to be able to achieve a frame rate of 15 fps.

There are also algorithms that use two or more images as inputs and are labeled as **multi-view** approaches. Huang et al. [10] developed a deep convolutional neural network to construct depth maps from a set of simultaneous[2] images (two or more) with a determined camera position and calibration. On the other hand, Ummenhofer et al. [11] use a multi-view approach not from images taken at the same time, but from two successive images.

[2] Taken at the same time

This technique is called structure from motion or motion parallax and produces a depth map based on the 3D information learned from the different perspectives in the two images.

Transfer learning is a technique that has also been used to tackle this computer vision problem. These approaches are not only useful for depth estimation and their efficiency and usefulness is demonstrated by the work of Zamir et al. [12]. In their work, they establish a relationship between different computer vision tasks and propose a convolutional approach to transfer learning from one context to another to reduce the need (and cost) of supervision-based techniques.

The **encoder-decoder** deep neural network architecture is used in some of the best performing methods not only in depth perception, but also for other computer vision related tasks including image segmentation, optical flow estimation, image restoration. In depth estimation this kind of neural networks have been used both for the supervised [13] and unsupervised [14] version of the problem. The encoder part of the network typically reduces the size of the input and produces a feature vector from the image while the decoder reconstructs the image (same size as the input) as a depth map based on the outputs of the hidden layer which varies between methods.

Monocular and one of the multi-view approaches (binocular) are further discussed in dedicated chapters (2 and 3). The encoder-decoder or autoencoder is one of the most common neural network architectures in state-of-the-art methods. On the other hand, transfer learning is present in those who achieve greater accuracy.

The most important benchmark for state-of-the-art evaluation of depth estimation methods is The KITTI Vision Benchmark Suite [15] which contains hundreds of methods ranked by accuracy. There are both binocular and monocular versions of the benchmark and a few of the methods have a related published paper and open source code. At the moment of turning in this thesis, these were the top 5 methods for both stereo and monocular versions of the solution.

BINOCULAR					
Rank	**Method**	**Open source**	**Paper**	**Error**	**Runtime**
1	M2S_CSPN	NO	[16]	1.74%	0.5 s
2	GANet-deep	YES	[17]	1.81%	1.8 s
3	AMNet	NO	[18]	1.84%	0.9 s
4	AcfNet	NO	N/A	1.89%	0.48 s
5	Samsung_System_LSI	NO	N/A	1.90%	0.4 s
MONOCULAR					
1	BTS	NO	N/A	2.21%	0.1 s
2	DL_61 (DORN)	YES	[19]	2.23%	0.5 s
3	DL_SORD_SL	NO	[20]	2.49%	0.8 s
4	BTS-256	NO	N/A	2.70%	0.1 s
5	DS-SIDENet_ROB	NO	[21]	2.87%	0.5 s

Table 1. Top 5 methods in the KITTI benchmark [15]

1.6 Regulatory framework

As a primarily pure research bachelor thesis not many regulations need to be taken into consideration, as opposed to a software development project. Nonetheless, in this section, relevant national regulation regarding present and future hypothetical work is presented and discussed.

1.6.1 Software

Since this research deals with the use of third-party algorithms and models, software licenses must be considered, especially when dealing with open source software. Contrary to proprietary software, owned by an individual or company, in open source software anyone can inspect, modify an enhance the source code [22]. This permission is given to an individual, company and/or organization via software licenses, which describe legal rights and obligations in regard to the use of that software.

GitHub projects are not open source by default and only by making the repository public (available for everyone to see and fork[3] it), does not immediately include a license. In this case, default copyright laws apply giving the author/s full rights to the source code, i.e. no one can reproduce, distribute or use it [23, 24]. Based on the project needs or the author's wishes there are several different open source licenses available. MIT, Apache 2.0 and GPLv3 are the most common software licenses and each one of them describe three scopes of conditions:

- **Permissions**: how you can use the source code (commercial use, modifications, etc.)
- **Conditions**: what the authors demand in exchange (disclose source, license and copyright notice, etc.)
- **Limitations**: what the authors do not provide (liability, warranty).

All the algorithms and models used in this research can be found in GitHub and thus have a license. DenseDepth uses the GPLv3 license, PSMNet uses the MIT license and Monodepth is owned by Niantic Inc. and uses the UCLB ACP-A license [25, 26, 27].

		Permissions		
Model	**License**	**Use**	**Modifications**	**Redistribution**
DenseDepth	GPLv3	YES	YES	YES
PSMNet	MIT	YES	YES	YES
Monodepth	UCLB ACP-A	YES	YES	NO

Table 2. Summary of repository permissions. Source: GitHub

These three models have repositories that license their software in a way that is useful to the purpose of this thesis but might need to be reviewed if they are required in a larger commercial project.

1.6.2 Cameras and footage

The use of this software may require recording images from a car. These images might show footage from people and/or places that can raise privacy concerns or even violate a law. In some countries, onboard cameras (dashcams) are commonly used to provide

[3] Make a copy of a repository, allowing the user to freely experiment without affecting the original.

evidence in case of accidents through continuous recording of external views and/or other data such as speed, steering or g-force [28]. Since there is already legislation about this type of image recording device, we can extend that same legislation and assume it covers the cameras required by the software that this thesis explores.

Different countries have contrasting legislation with regard to dashcams and data protection that go from no punishment (Russia) to expensive fines or actual imprisonment (Luxembourg). In Spain, where this research is taking place, there are two main laws that need to be considered: LOPD (Organic Law on Data Protection) and DGT (Directorate-General of Traffic) regulations.

General Traffic Regulation does not refer to dashcams or any other kind of recording device specifically, but Article 18 [29] requires drivers to ensure the necessary field of view and full attention to the road to guarantee not only their safety, but also the safety of other passengers, pedestrians and other vehicles. It also states that using devices such as TV screens, video players, etc. are incompatible with the required driving attention. Therefore, dashcams are not regulated, meaning they are not illegal, but cannot be manipulated while driving.

LOPD, on the other hand, does not forbid the recording of public places. Article 22 [30] allows footage of public places for video surveillance purposes but compels the user to delete it after one month This footage should not be uploaded to the internet. Consequently, video footage recorded continuously from a camera in a car is legal as long as it is for private or video surveillance use.

1.7 Socio-economic environment

In this segment a breakdown of the project budget is presented, followed by an analysis of the possible impact in society and the economy. The last part mentions the ethics involved in projects like this and comments on a few of the issues that may arise from data mishandling and what guidelines are useful to implement.

1.7.1 Project budget

This project was part of a research grant sponsored by BQ through their foundation Deep Technology & Engineering Services, S.L. The scholarship is scheduled to last 8 months and started last November 2018. The grant consisted on financial aid that was distributed monthly to the student and author of this thesis and the head of research, who is also the thesis tutor. Details and a breakdown of the project budget are detailed in the following epigraphs.

1.7.1.1 Staff

The gross monthly grant allowance for the research student was 500,00 EUR and involved a 20 hour work week, resulting in approximately 6,25 EUR per hour. The head of research contributed in weekly meetings of 2 hours of duration, which sums up to 32 hours. His salary is not publicly available, but it is estimated according to the Spanish Ministry of Labour and its collective agreement for engineering companies' salaries [31].

Position	Name	Hourly rate	Hours	Total
Head of research	Miguel A. Patricio Guisado	13,12 €	64	839,68 €
Research student	Daniel Sarmiento Rocha	6,25 €	640	4000,00 €
Total				**4839,68 €**

Table 3. Staff budget summary

1.7.1.2 Hardware & Software

The hardware used in this research consisted of an HP Pavilion laptop computer whose specifications are detailed in Chapter 4. Most of the software used in this project is open source and its use is free of charge. Hardware and software costs are adjusted based on the following amortization formula:

$$\text{Total cost} = \frac{\text{Purchase price}}{\text{Amortization period}} \times \text{Usage period} \qquad (1.1)$$

Description	Purchase price	Amortization[4]	Usage	Final cost
Laptop	799,00 €	13,32 €	8 months	106,56 €
Total hardware				**106,56 €**
Windows 10 Home	69,99 €	1,17 €	8 months	9,36 €
TensorFlow	0,00 €	0,00 €	6 months	0,00 €
PyTorch	0,00 €	0,00 €	6 months	0,00 €
Other software licenses[5]	0,00 €	0,00 €	6 months	0,00 €
Total software				**9,36 €**
Total				**115,92 €**

Table 4. Hardware and software budget summary

1.7.1.3 Budget summary

Next up is a table summing up the project budget detailed previously. Additionally, the VAT (IVA in Spain), has been added and its rate is 21% according to the Spanish general tax legislation since 2012 [32].

Description	Cost
Staff	4839,68 €
Hardware	106,56 €
Software	9,36 €
Total before taxes	**4955,60 €**
Taxes (21%)	1040,68 €
Total	**5996,28 €**

Table 5. Budget summary including taxes

1.7.2 Socio-economic impact

This thesis revolves around many fields and areas that may be sensitive to the impact of the implementations of each individual or organization. In this section, these elements are presented and discussed.

[4] Monthly
[5] PSMNet, Monodepth and DenseDepth algorithms open sourced in GitHub

1.7.2.1 Economy and society

As neural networks and artificial intelligence become more accurate and efficient, they are switching from the academic and scientific world and are starting to impact the whole economy. These improvements can lead to many that both businesses and consumer can enjoy. Better customer analysis/segmentation, medical diagnosis and smart recommendations are just a few examples. This research is about one of the many cases in which neural networks can make a difference.

Depth maps from monocular cameras can contribute to make autonomous driving more affordable as it promotes independence from more expensive hardware. This does not only refer to traffic (which also includes public transportation and freight traffic), but also agriculture, storage management and any other means of transportation subject to automation. Automation is an issue itself because it might lead to change in which jobs are performed by humans and which ones are better tasks for robots. However, this particular case, does not seem to impact society in that way as it affects things that are already automated.

1.7.2.2 Ethics

Another key aspect that must be considered is ethics. Artificial intelligence has always raised some ethical issues and machine learning as a field is not exempt from that too. Neural networks, like any other learning algorithms, are sensitive to the data that is fed to them by humans during training. Recently, this has been a growing issue that needs to be addressed as it may lead to some degree discrimination [33]. To avoid this, it is recommended to follow ethical guidelines like the Responsible Machine Learning Principles [34] by the Institute for Ethical Machine Learning when developing an AI project.

Chapter 2

Binocular vision and Stereo Matching

2.1 Binocular vision

In biology, binocular vision is the type of visual perception present in every animal with two eyes that do not work independently. This is an advantage in many aspects and one of them is the possibility of stereoscopic depth perception [35]. Having two eyes, especially when both are facing directly forward, results in some degree of image overlap. The horizontal separation between eyes is what provides the brain with two different images that have a portion of the image information in common, this is called binocular overlap. However, this horizontal separation also results in horizontal disparities between both images (binocular disparities). In other words, the relative position of two objects that are separated in depth from the viewer will be different in the two eyes. These disparities are then used by the visual cortex of the brain to infer depth in a process called stereopsis [36].

This topic has been researched way before our times by numerous scientists like René Descartes in the 17th century, which illustrated how distance was perceived using binocular vision [37]. In the field of computer vision, we can imitate this biological process using two cameras that take two simultaneous pictures and a stereo matching algorithm that mimic the eyes and visual cortex respectively.

2.2 Stereo matching

Stereo matching or stereo correspondence is one of the tools used in computer vision to extract depth information from a pair (generally) of left and right images. It consists in finding which points in both images are projections of the same scene point in the physical world [38]. These two pixels with coordinates (x, y) and (x', y) differ in what is defined as disparity or the difference between their horizontal values $d = x' - x$. Due to the geometry of most stereo cameras (parallel optical axes) and known focal length[6] (f) and baseline[7] (b) it is possible to determine per-pixel depth via triangulation [39]:

$$z = \frac{f \times b}{x_L - x_R} = \frac{f \times b}{d} \qquad (2.1)$$

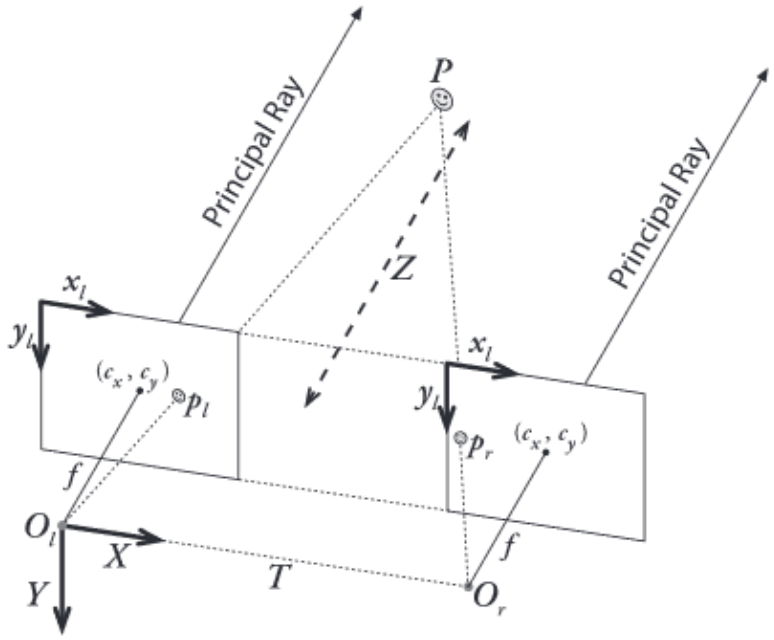


Figure 3. Epipolar geometry for stereo matching with rectified cameras [40]

Since disparity (d) is inversely proportional to depth (z), a disparity map will be inversely proportional to a depth map. That is, more disparity will be obtained from those points who are closer to the camera than from those who are far. We can empirically observe this phenomenon by placing a finger right in front of our eyes, closing one eye at a time

[6] Distance from where light rays converge to the digital sensor
[7] Distance between cameras, degree of separation

and then the other. Our finger appears to be moving more than other objects that are further from our eyes; i.e. more disparity. Since disparity and depth are correlated, instead of directly extracting the depth from the image pair, we can now estimate the disparity map, a monochromatic image in which brighter pixels represent more disparity (less depth) and darker pixels less disparity (more depth).

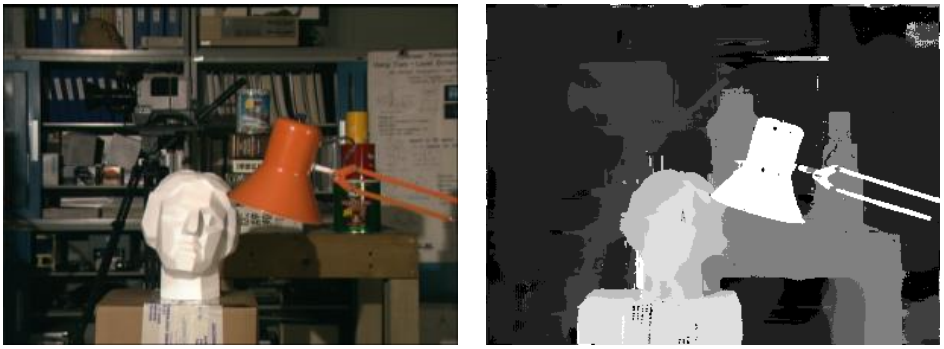


Figure 4. Example left image input (left) and disparity map (right) [41, 42]

However, stereo matching has its disadvantages and problems dealing with some pixels in the image due to a couple of main difficulties [43]:

1. **Ambiguity**: When there are many pixels of nearly the same color, the correct match may not always be the one with less difference to the target pixel. This is especially problematic with large textureless areas, containing many pixels of almost the same color.
2. **Occlusion:** A point in the physical world is represented in one image but not in the other due to the horizontal separation between cameras and the change in perspective. This point does not have a correct match in the other image because it does not exist there. If the algorithm finds a matching pixel, it would create errors in the depth map.

2.2.1 Stereo matching approaches

Stereo matching algorithms can be classified into local or global approaches [44]. The former considers only local regions of information (window) for each pixel, making it less computationally expensive. The emphasis in these approaches is on the cost aggregation steps. Global methods on the other hand, produce better results but have greater computational complexity.

The majority of stereo matching algorithms perform the following four steps [45] to get from a pair of input images to a disparity map.

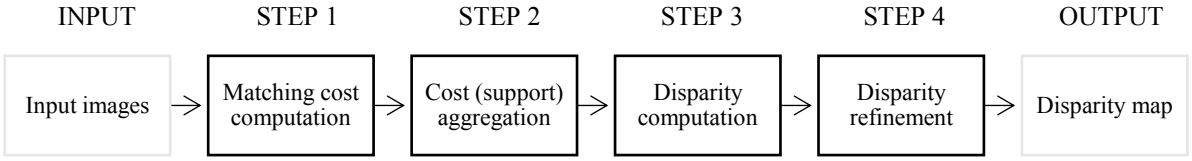


Figure 5. Steps of a typical stereo matching algorithm. Source: [45]

A brief overview of each of the steps is provided below:

- **Matching cost computation**: the degree of similarity (matching cost) is computed at each pixel for all pixels considered depending on constraints relative to the method (local, global).
- **Cost aggregation**: critical stage for local methods.
- **Disparity computation**: central stage for global methods.
- **Disparity refinement**: reduction of errors caused by noise, and therefore improvement of the disparity map.

2.2.2 Stereo matching performance

Performance of stereo vision algorithms, along with other computer vision related tasks, is evaluated in many benchmark suits like Middlebury and KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute). In this thesis, the latter was selected because it provides a more realistic and appropriate dataset for the research topic. KITTI Stereo 2015 consists of 400 scenes, 200 for training and 200 for testing. These scenes contain moving objects, such as cars, trucks, bicycles and pedestrians, captured with a stereo camera by driving around downtown Karlsruhe (Germany), as well as rural areas and highways. The ground truth is captured by a Velodyne Laser Scanner (LIDAR) and GPS. An example of the two input images and ground truth is shown in the following table.

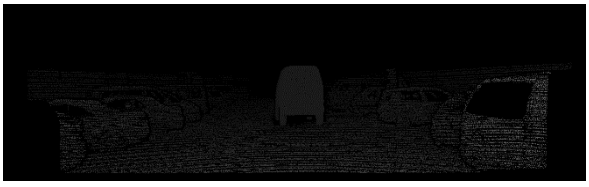
Left image	
Right image	
Ground truth	

Table 6. Example of a KITTI 2015 dataset training trio

There are a few things that are worth mentioning about this benchmark. At the moment of writing this thesis, there were a total of 197 listed methods in the KITTI Stereo Benchmark, 38 of which have also made the code public (19.29%) and 75 have a related published paper (38.07%). The top 10 performers in this benchmark achieve an average error of 1.90% and a runtime of 0.61 seconds. The top performer stands at 1.74% and 0.5 seconds. The fastest method completed the task at 0.02 seconds with 3.08% error. This is relevant because some applications need this task to be solved in real time.

	Method	**Error (less is better)**	**Runtime (less is better)**
Most accurate	M2S_CSPN [16]	**1.74%**	0.5 s
Fastest	Fast DS-CS	3.08%	**0.02 s**
Reproduced	PSMNet [46]	2.32%	0.41 s
	TOP 10 Average	1.90%	0.61 s

Table 7. Performance summary of stereo matching methods in KITTI benchmark

As the KITTI's benchmark results show, stereo vision approaches have very decent performance. However, they are limited by the baseline distance between the two cameras, which causes the depth estimates to be less accurate with larger distances [47]. Nearly all of the best performing methods use Deep Learning, specifically Deep Convolutional Neural Networks.

2.3 Pyramid Stereo Matching Network

Although binocular stereo matching is not part of the main focus of this work, one of the few open sourced models ranked in the KITTI benchmark was chosen to be reproduced and analyze its results to compare them to the monocular-based methods that were part of the experiments in Chapter 4. Pyramid Stereo Matching Network or PSMNet consists of two Spatial Pyramid Pooling (SPP) siamese components whose output is then fed to a stacked hourglass module. An overview of the architecture can be seen in the figure below.

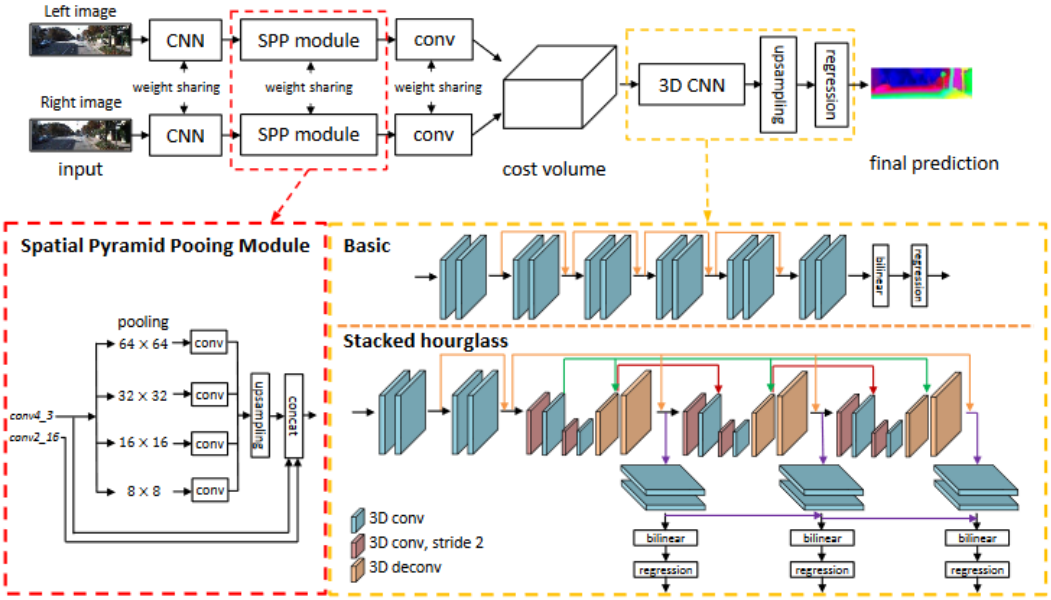


Figure 6. Architecture overview of PSMNet [46]

As mentioned in the previous section, PSMNet achieved state-of-the-art precision on the KITTI benchmark. An example of one of the results obtained on the KITTI dataset by the authors is shown in the following figure. The colormap used by the authors represents high depth values with dark blue and low depth values with a bright green, reddish for middle distance.

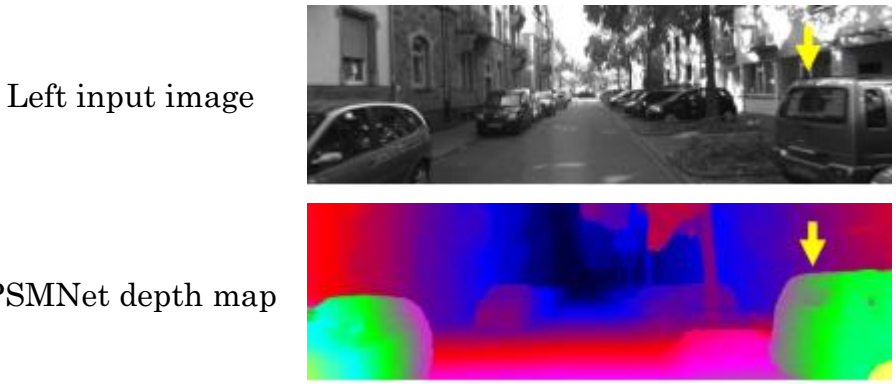


Figure 7. Example of results of depth estimation for PSMNet [46]

The corresponding depth map displays good results that extract correct 3D information from the scene and appears to preserve sharper object boundaries than other stereo matching methods. However, attempts to replicate their results in this thesis were not successful and therefore are not part of the experimental chapter (Chapter 4). The figure below shows an example of a PSMNet output on the KITTI dataset, the same as the one used in training. Details on the environment running the neural network are explained in the first section of Chapter 4 but should not influence the model's accuracy.

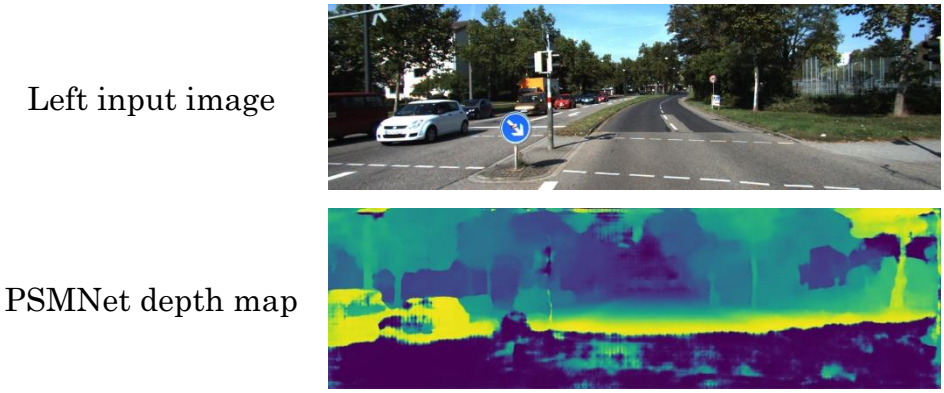


Figure 8. Attempt at reproducing PSMNet results on KITTI

Looking at the resulting depth map it is easy to perceive that there is something wrong with the area that is closest to the camera. It gives wrong disparity values and fuzzy edges up to a certain distance where it starts representing real world depth. According to the color map described below, there are high depth values (purple) both right in front of the camera and at the back, where they should be that way.

Figure 9. Viridis color map used in PSMNet outputs

It is possible that the best pretrained model made available by the authors was not that great or that the model needed preprocessing or postprocessing to give decent results. However, retraining the network was not possible due to memory constraints as these neural networks with siamese modules are very hardware demanding. I contacted the author via email, trying to clarify if anything was wrong with my version of their implementation and to ask about pre and postprocessing. Unfortunately, their recommendations did not change the output.

Chapter 3

Monocular vision approaches

3.1 Monocular vision

Monocular vision may refer to the type of visual perception in binocular animals that have lost vision in one eye or the one present in animals that use each eye independently. For the task of judging depth from a single image, e.g. when we look at a picture and not the real scene, humans make use of monocular cues such as texture variations and gradients, occlusion, known object sizes, haze, defocus, etc. [47]. In other words, human beings can perceive the depth of a scene (three-dimensional element) from a picture of that scene (two-dimensional element). Therefore, monocular vision and monocular cues may be enough to recognize depth.

3.2 Monocular depth perception

Binocular depth perception is an interesting take on this task because it relies on less expensive hardware than other methods using radar, LIDAR or ultrasound sensors. However, there are still two cameras and a series of conditions that are needed for stereo matching algorithms to work properly. Monocular depth perception avoids all of these constraints and is an even better choice for smaller, less powerful devices since it only

uses one single camera. Not only because a one camera setup is less bulky, but also because it would be less power intensive.

3.2.1 Monocular depth perception approaches

Approaches using binocular depth perception implemented stereo matching algorithms in their methods because their input information came from a pair of stereo images from a left and right camera. This is not the case in monocular depth perception, where the input is usually a single image. Some other approaches use displacement of devices with one camera to estimate depth [48]. This phenomenon is called parallax and is also a visual cue in biological vision. However, since these approaches use two images (although not from two cameras, but from two different points of view and at different times), will not be taken into consideration in this thesis.

3.2.2 Monocular depth perception performance

The KITTI Vision Benchmark Suite has also a variant for depth prediction evaluation from a single RGB image and consists of 93000 training, 1000 evaluations and 500 test images. At the moment of writing this paper, there were 31 submissions listed in the Single Image Depth Prediction benchmark, with only 3 of them making the code public (9.68%) and 7 are based on a published research paper (22.59%). The top 10 performers accomplish an average error of 2.63% and an average runtime of 0.369 seconds. The top performer and the fastest method stand at 2.21% error and 0.01 seconds, respectively.

	Method	**Error (less is better)**	**Runtime (less is better)**
Most accurate	BTS	**2.21%**	0.1 s
Fastest	MultiDepth	3.89%	**0.01 s**
	TOP 10 Average	2.63%	0.369 s

Table 8. Performance summary of monocular depth in KITTI benchmark

The KITTI Single Image Depth Prediction Benchmark shows that, despite having significantly less listed methods than the Stereo Vision Benchmark, monocular depth perception is still an interesting and valid approach. Stereo vision methods are more accurate, but slower, which might make monocular approaches more valuable for tasks and/or environments in which speed, rather than flawless accuracy, is preferred.

3.3 Selected models

For the experimental section, in the next chapter of this thesis, two monocular approaches have been selected in addition to the stereo vision approach shown in the previous chapter. By the time this section is being written, neither one nor the other are currently listed in the KITTI Single Image Depth Prediction Benchmark. These two selected methods are called Monodepth and DenseDepth.

3.3.1 Monodepth

Monodepth is a method based on the published paper *Unsupervised Monocular Depth Estimation with Left-Right Consistency* [14]. Monodepth approaches the depth perception task as an image reconstruction problem, i.e., finding a function to predict per-pixel depth from a single image, $\hat{d} = f(I)$. This method avoids the use of LIDAR ground truth data in training, arguing that capturing reliable depth data is expensive, time consuming and might produce unreliable depth data for moving and/or transparent objects. Other sensors, like the active light sensors, such as Microsoft Kinect, have very limited range and perform poorly outdoors.

Left image

Ground truth depth data (from LIDAR)

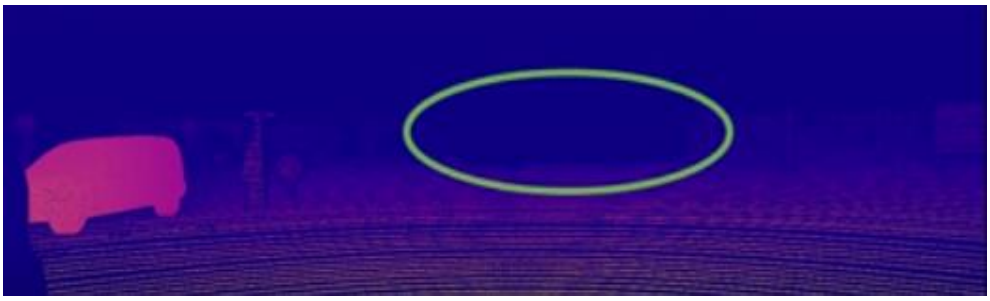


Figure 10. Limitations of LIDAR. Bus is ignored as it is a moving object. [14]

Instead of using two images as input and ground truth data as in a supervised approach, Monodepth aims to learn to reconstruct an image (left or right) from the other during training, thus learning about the three-dimensional structure of the scene. In other words,

before trying to estimate depth, this approach finds a dense correspondence field d^r that produces an artificial right image from the left one (or a left image from the right one).

$$\tilde{I}^r = I^l(d^r) \quad (3.1)$$

$$\tilde{I}^l = I^r(d^l) \quad (3.2)$$

Both disparity maps (left and right) are predicted at the same time and forced to be consistent with each other to obtain *d.* If the images are rectified and the baseline distance *b* and the focal length *f* are known values, the depth $\hat{d}$ can be calculated from the disparity and the stereo equation:

$$\hat{d} = \frac{b \times f}{d} \quad (3.3)$$

Monodepth uses a convolutional neural network (CNN) to estimate both disparity maps (left-to-right and right-to-left) from a left input image and its inspired in the DispNet architecture. The authors do not provide a graphical overview of the network architecture in their paper. Training takes 25 hours in a Titan X GPU on 30,000 images for 50 epochs. Inference (estimating depth from an image) takes about 35 ms for an image with a 512×256-pixel resolution. Assuming each image takes 35 ms to process, we can infer that a rate of ~28.57 frames per second can be achieved. Its performance in the KITTI Stereo Benchmark is not remarkable, achieving a 23.81% of error (which would rank them at 198). As a side note, they also developed a stereo version which uses two images as input and outperforms the monocular version with a 9.19% error.

Although Monodepth can infer depth from a single image, it still needs a pair of rectified stereo images during the training stage of learning. However, it is worth noting that this training is less expensive because less hardware is involved and enables the use (and creation) of datasets without needing expensive ground truth depth data.

3.3.2 DenseDepth

DenseDepth is a much more recent method based on the paper *High Quality Monocular Depth Estimation via Transfer Learning* [49]. DenseDepth relies on transfer learning, a process that uses previous knowledge derived from a learning problem —image

classification in this case— to help solve another —depth estimation— more efficiently [50]. Transfer learning allowed this method to provide a simpler and modular architecture with similar or even better results than other methods.

It can be argued that DenseDepth is a "more monocular" approach than Monodepth because it takes only an input image both in training and prediction/inference. However, it is a fully supervised learning method that needs ground truth depth data such as those from 3D laser scans or other expensive hardware.

Their network architecture follows an encoder-decoder structure. The encoder is where the transfer learning occurs, specifically using DenseNet-169 pretrained on ImageNet [51], an image database for image classification and object recognition. Having some pretrained section allowed this method to reduce validation loss compared to a completely random weights initialization. The decoder section is composed of basic convolutional layers.

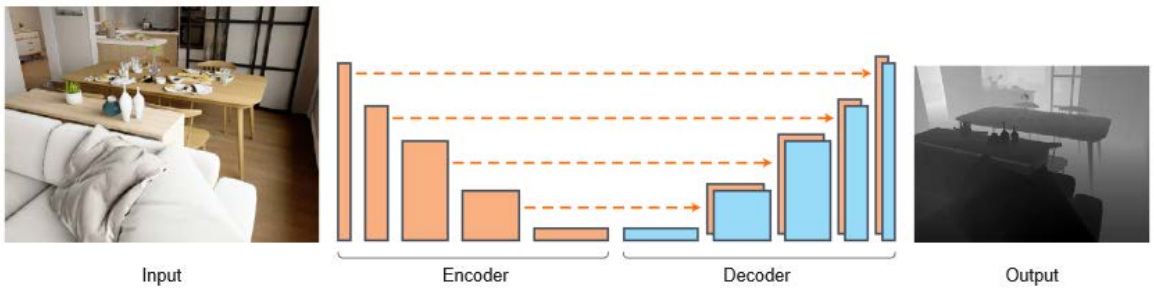


Figure 11. Architecture overview of DenseDepth [49]

One of their considerations to avoid overfitting is the use of a procedure called data augmentation. In the case of image data augmentation, this technique consists mostly in the use of transformations such as rotating, flipping and/or cropping input images [52]. However, there are other methods such as using style transfer to change the input image from night to day, winter to summer, sunny to cloudy and vice versa. This technique increases the amount and diversity of data a machine learning algorithm is fed, thus improving its performance and judgment. Due to the nature of the task in DenseDepth, the only valid data augmentation transformation is the horizontal flipping of images. Other transformations that change the geometry of the scene are counterproductive. For example, a vertical flip would put the road and cars at the top of the scene, while the sky would be at the bottom, a situation that would not match any future inference input image, nor any

real world situation. The following table includes examples of some of these transformations for clarification.

Method	Input image	Output image
Deep Style Transfer (seasons)		
Deep Style Transfer (time of day)		
Horizontal flipping		
Vertical flipping		

Table 9. Illustration of Data Augmentation techniques [53]

In DenseDepth, the mirrored images (horizontal flipping) are used to average the depth values of the original image with its horizontally flipped version.

Training took 20 hours for the NYU Depth v2 [54] dataset and 9 hours for the KITTI dataset on four NVIDIA Titan Xp. Its performance in the NYU Depth v2 dataset is remarkable, achieving top accuracy in most of the metrics (89.5-99.6%) which makes it state-of-the-art. Its performance in the KITTI dataset is not as remarkable, but still great with an accuracy of 88.6% to 98.6%.

Chapter 4

Experimental results

4.1 Environment

For the purpose of context and reproducibility an overview of the environment in which the experiment was conducted is stated next. Both models were run on the same laptop computer on an NVIDIA GeForce 940m graphics card (4 GB of VRAM). Monodepth was run using TensorFlow [55], a software library for machine learning and neural networks developed by Google. The second model, DenseDepth was run using PyTorch [56], also a machine learning and neural networks library developed by researchers at Facebook.

Hardware environment	
CPU	i7-6500U @ 2.50 GHz
GPU	GeForce 940m
VRAM	4 GB
RAM	16 GB
Software environment	
Programming language	Python 3.5+ with Anaconda
Framework (I)	TensorFlow 1.13.1
Framework (II)	PyTorch 1.1.0

Table 10. Summary of the operational environment

4.2 Pre-trained models

The original implementations of both Monodepth and DenseDepth provide open source pre-trained models for everyone trying to replicate their experiments. That is, training has already been done on large datasets by the authors and a model has been selected as the best performing for a given dataset. With a pre-trained model, the best weights can be applied to the network so that the depth inference can be computed without the need of retraining the network, a very demanding process due to the high complexity of convolutional neural networks architecture.

Monodepth has six different pre-trained models available. The first three are trained on KITTI, Eigen and Cityscapes datasets. The next two are trained on Cityscapes and fine-tuned[8] on KITTI and Eigen respectively. The last one is a stereoscopic variation of the network trained on the KITTI dataset. On the other hand, DenseDepth only provides two different pre-trained models: one trained on the NYU Depth V2 dataset and the other trained on the KITTI dataset. A summary of the available pre-trained models is provided in the following table:

	Name	**Dataset**	**Description**	**Fine-tuned**
Monodepth	*model_kitti*	KITTI	Exterior, city road	NO
	model_eigen	EIGEN	Interior, rooms	NO
	model_cityscapes	Cityscapes	Exterior, city road	NO
	model_city2kitti	Cityscapes	Exterior, city road	KITTI
	model_city2eigen	Cityscapes	Exterior, city road	EIGEN
	model_kitti_stereo	KITTI	Exterior, city road	NO
DenseDepth	*nyu*	NYU Depth V2	Interior, rooms	NO
	kitti	KITTI	Exterior, city road	NO

Table 11. Summary of available pre-trained models.

Pre-trained models are expected to perform better extracting depth from images taken in similar environments as the dataset that was used to train the models. For example, for the task of depth perception in an autonomous vehicle navigating through a city, it would

[8] Like transfer learning, taking advantage of a previous solved task to initialize weights

be better to use the models trained with KITTI or Cityscapes. However, for 3D reconstruction or navigation of interiors, EIGEN or NYU Depth V2 are more suitable.

4.3 Results

In chapters 2 and 3, Monodepth and DenseDepth precision was discussed and presented. Those accuracy tests used ground truth data available for the images in the dataset to evaluate the precision of each method. However, in this section a quantitative analysis is not possible because the images that are being used do not have corresponding ground truth data. This allows for a qualitative analysis of both Monodepth and DenseDepth using images that they have never seen before as they do not belong in any dataset visible to them during training. Images were randomly selected and due to DenseDepth limitations they are all 640x480 resolution. These input images are half from indoor spaces (bedrooms, kitchens, living rooms, etc.) and the other half from roads and cities taken from the perspective of a driving car. This lets both methods showcase their performance in two main different scenarios. Images are taken from stock photos and/or are free to use, not commercially.

4.3.1 Monodepth

Monodepth has six pre-trained models available which were trained and fine-tuned with KITTI, Cityscapes, Eigen or a combination of them. The depth/disparity maps obtained with this model are color-coded using the plasma color map described below.

Figure 12. Plasma color map used in DenseDepth outputs

The two best performing for each of the test cases (indoor images and outdoor images of roads and cities) were selected and their results are presented below. For the indoor images, the best results were obtained using *model_city2eigen*, trained with Cityscapes and fine-tuned with EIGEN.

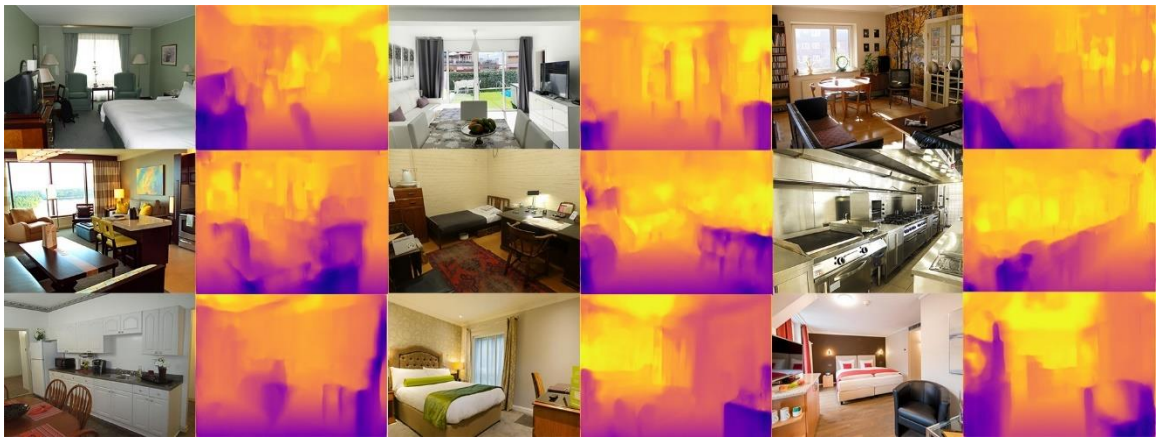


Figure 13. Results from Monodepth using city2eigen pre-trained model

Depth maps from this method are fuzzy and it is hard to even match each output to its corresponding input, which means that some important features are lost in the process. Object boundaries are not clear enough, but some degree of depth and perspective information matches real world depth from the input. At best this is a coarse, but not dense depth map.

The best performing model for roads and cities images was model_city2kitti, trained with Cityscapes and fine-tuned with KITTI.

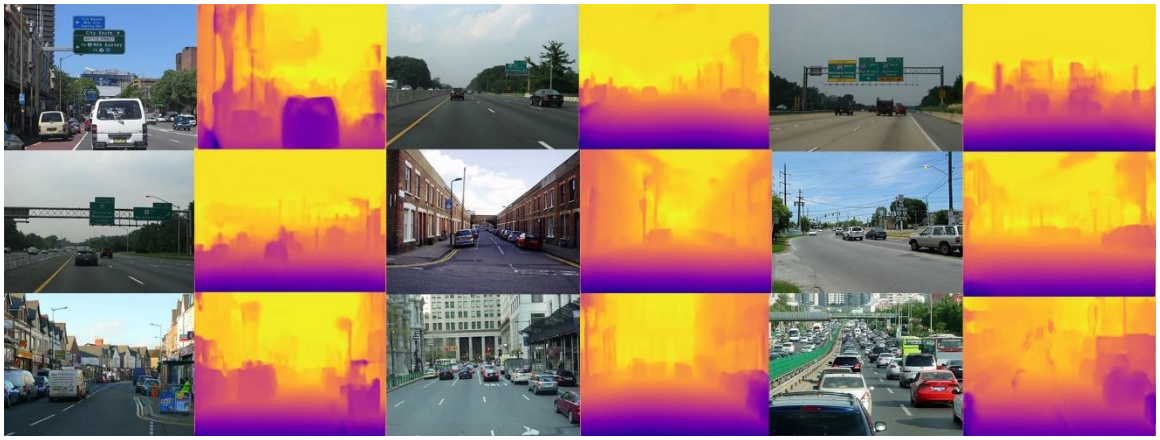


Figure 14. Results from Monodepth using city2kitti pre-trained model

Although way better than the results from indoor images, Monodepth results are far from remarkable. It does a decent work at identifying and extracting depth from roads and some of the objects, especially when they are at close to medium distance. Nonetheless, it fails at detecting cars which are close to the camera in the bottom right image, mixing them with the road.

Monodepth does not perform as well when using some of the other pre-trained models. Those results are shown in Annex A of the Appendix.

4.3.2 DenseDepth

DenseDepth only has two pre-trained models available, one using KITTI dataset and the other using NYU v2 dataset. The best results were obtained when the NYU pre-trained model was used to extract depth maps from indoor images and when the KITTI pre-trained model was used with images from roads and cities. The following figure shows the best depth maps obtained from the random sample of indoor images, along with the corresponding monocular inputs.

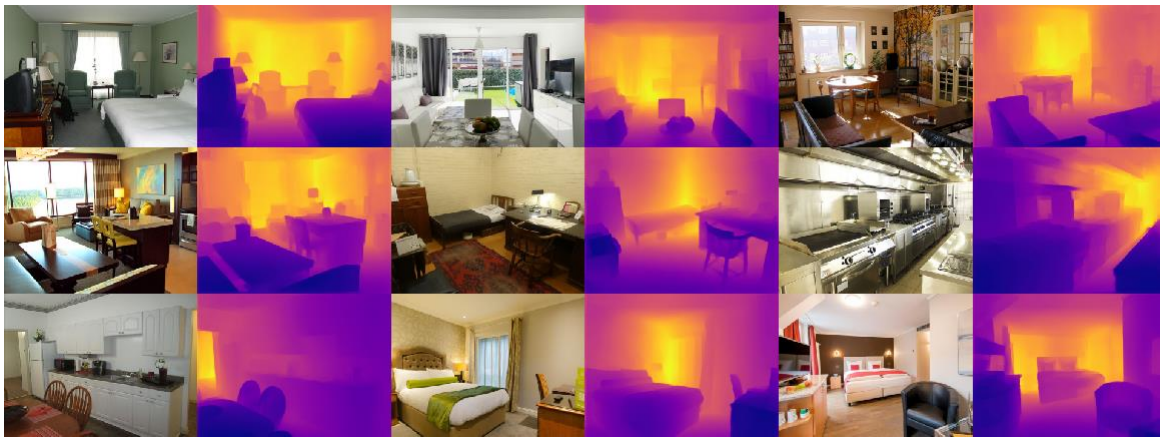


Figure 15. Results from DenseDepth using NYU pre-trained model

DenseDepth is able to extract depth information from these monocular indoor images with a high level of detail, which can be observed in many features like object boundaries and the many levels of real-world depth it shows.

The model pre-trained with data from the KITTI benchmark was tested on random images from roads, highways and cities. In the following figure the best results from this experiment can be observed and compared with the monocular input image that was fed to the algorithm.

Figure 16. Results from DenseDepth using KITTI pre-trained model

Again, DenseDepth does a great work at inferring 3D information from a monocular input image. In this case, the model's performance is best shown in the images with many different cars at different distances from the camera. It not only displays how many different (and well defined) objects are (cars), but also which ones are further and closer from the observer. However, it has problems evaluating real depth from the sky, which causes it to confuse it for ceiling, creating depth maps that sometimes resemble tunnels.

This method performs significantly worse when it uses the KITTI pre-trained model with indoor images and the NYU pre-trained model with roads and highways images. Those results are shown in Annex B of the Appendix.

4.4 Comparison

DenseDepth performance in indoor scenarios is significantly better than Monodepth's. It is not hard to match each DenseDepth output to its input because object boundaries and depth information is clear, unlike in Monodepth. The following figure shows a few examples of the difference between the outputs produced by the two methods compared to their input images.

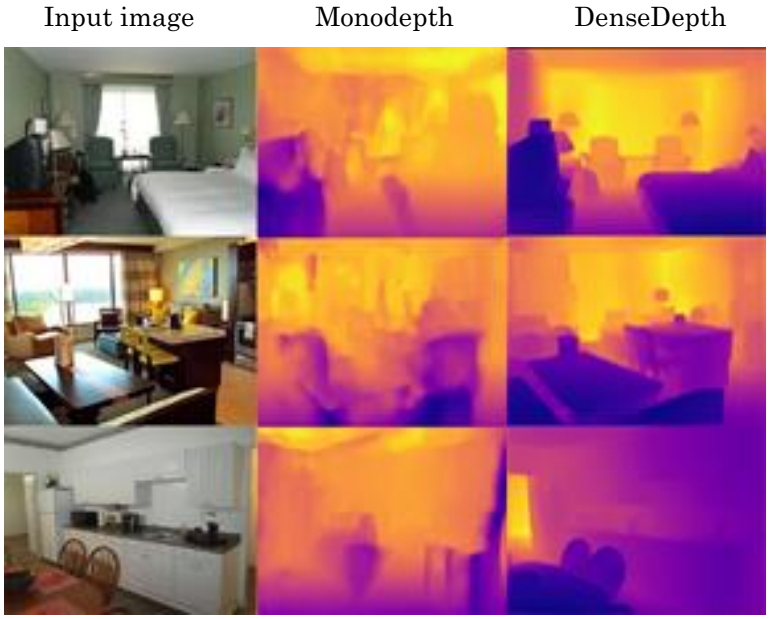


Figure 17. Comparison of indoor depth maps (Monodepth vs DenseDepth)

The case with outdoor scenarios regarding roads and cities is not as clear as the previous one. Monodepth seems to be more sensitive and produce a more progressive depth map,

which can be observed mainly on the road itself, but misses important objects like cars (bottom input image). This seems to be a problem derived from the issue of dealing with occluded pixels explained in Chapter 2. DenseDepth does not always represent true depth from the road but gets every object in the picture right, even transparent or translucid pieces like car windshields.

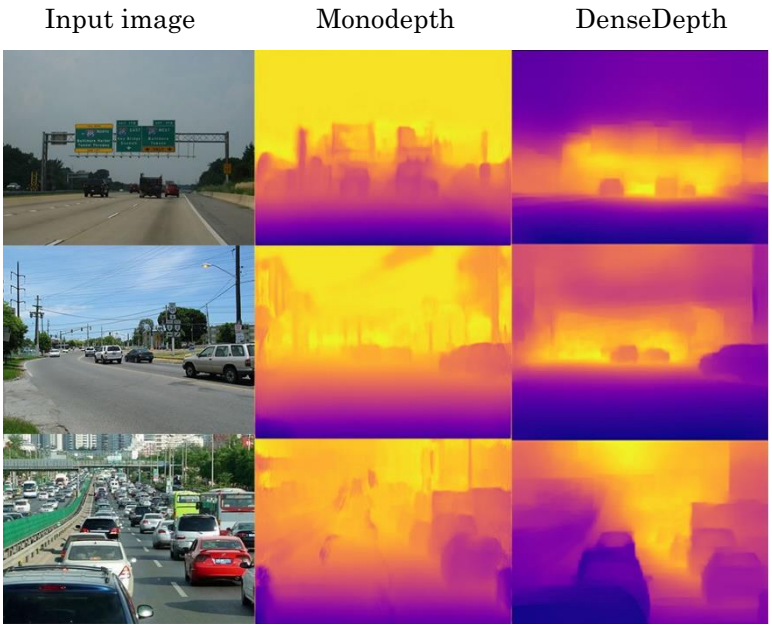


Figure 18. Comparison of roads depth maps (Monodepth vs DenseDepth)

These results resemble the accuracy metrics described in Chapter 3 for both Monodepth and DenseDepth. Monodepth had performed worse than DenseDepth on benchmarks and it did the same in the experiments described in this chapter. Time efficiency and performance metrics were discarded in these experiments because of hardware limitations. The amount of GPU VRAM available was not enough for these two methods to perform at their best and it was not consistent between the two of them.

Chapter 5

Conclusion and Discussion

5.1 Conclusion

Although binocular-based algorithms seem to be more straightforward when imitating animal and human vision to extract 3D information, monocular-based methods are also a valid approach nowadays and they seem to keep on improving. It has not only been proven to be a valid approach, but also a decent alternative, especially when constraints prevent the use of two cameras.

After reproducing both Monodepth and DenseDepth, the conclusion is that the latter achieves better qualitative results in unknown data (images that has not seen before). This corresponds to the quantitative results reported by the authors of both methods that were discussed in Chapter 3. However, the two of them were able to output comprehensive depth maps, which was one of the objectives, that could be used for projects that benefit or depend on this information. Nevertheless, the accuracy and consistency of these two algorithms (and monocular approaches in general) might not be enough for demanding products, especially those regarding human safety such as autonomous vehicles. Taking this into consideration, monocular depth perception can still be used as a complement to other sensors or as a part of a redundant system for this kind of products.

After this thesis, the importance that the training domain has is very clear. Every learning-based solution, whether machine learning or deep learning, should pay very close attention to the training data and its collection. Chapter 4 confirmed this by showing how

different the results from different models were, depending on which data was used during training and/or fine-tuning of the neural network. This is often an overlooked issue in deep learning, where the majority of the focus is usually on the neural network architecture and its layers rather than on the data that is going to feed the algorithm and create the model. Choosing and collecting proper training data is especially difficult when this data is in the form of images, instead of purely numerical values, because it is harder to be certain about when the whole spectrum of a problem or task is covered. If, for example, one of the inputs of a neural network is GDP[9] per capita, it is easy to research for data covering cases across the whole range of GDP per capita in the world. On the other hand, with images, there is no *known range* and it is problematic to cover all the cases.

To sum up, the main goal (obtaining a depth map from a single RGB image) was achieved. It was possible thanks to the other subobjectives developed and completed during the writing of this thesis. In the next and final section, some possible future lines of further research related to monocular depth perception are presented and discussed based on the results obtained during this study.

5.2 Discussion and future work

This field of research is as interesting and useful as lucrative. Many different companies are working on depth estimation for a wide variety of purposes. Some may use it as a way of adding filters (e.g. *bokeh*) to pictures and selfies (portrait mode) depending on the distance to the camera, while some might use it as a way to substitute or complement other sensors in autonomous vehicles, both aerial and terrestrial. Since this is a profitable research and these companies compete between them, not many of the best performing algorithms and methods have been open sourced. As noted in Chapters 2 and 3, the amount of open source code of the methods ranked in KITTI benchmarks is less than 20% for stereo vision and less than 10% for monocular depth estimation.

The field of computer vision is constantly evolving, improving and finding new approaches, solutions and applications for the field's most interesting tasks. The problem of

[9] Gross Domestic Product

extracting 3D information from a single image is a fascinating line of research for future related work.

Since both Monodepth and DenseDepth were run on a computer GPU, models did not need to be especially compact or light. However, one of the most appealing lines of work seems to be adapting the model so that it can run easily on a mobile device. This could be developed using *Qualcomm Neural Processing SDK for AI*, which allows neural networks trained in Caffe, ONNX or TensorFlow on Snapdragon mobile devices [57].

As mentioned earlier, choosing the right pre-trained model determined the quality of the depth maps that both methods produced. Another future approach would be related to domain-specific training. This thesis did not involve retraining the models, but it would be interesting to see their performance after training on other domains not present in the pre-trained models like portraits (people's faces), large field of view landscapes or darker scenes.

By the time of finishing this thesis, a second version of one of the selected methods, Monodepth v2, was released and made open source with the same licenses as its predecessor. Their authors argue that it is significantly better than the previous version and its competitors, achieving state-of-the-art results in the relevant benchmarks [58]. With this into account, it would be interesting to explore the possibilities of this much more recent version of Monodepth.

.

References

[1] O. Moindrot, "CS 131 Lecture 1: Course Introduction," *Stanford University.*

[2] K. Ikeuchi, Computer Vision: A Reference Guide, Springer, 2014.

[3] N. Lazaros, G. Christou Sirakoulis and A. Gasteratos, "Review of Stereo Vision Algorithms: From Software to Hardware," *International Journal of Optomechatronics,* vol. 2, no. 4, pp. 435-462, 2008.

[4] E. LeDell, "Deep Learning in H20," 10 October 2016. [Online]. Available: https://github.com/ledell/sldm4-h2o/blob/master/sldm4-deeplearning-h2o.Rmd. [Accessed 10 June 2019].

[5] A. Karpathy, "Convolutional Neural Networks," *CS231n Convolutional Neural Networks for Visual Recognition,* 2018.

[6] deeplizard, "Convolutional Neural Networks (CNNs) explained," Machine Learning & Deep Learning Fundamentals, 9 December 2017. [Online]. Available: https://deeplizard.com/learn/video/YRhxdVk_sIs. [Accessed 10 May 2019].

[7] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *ICLR*, 2015.

[8] D. Eigen, C. Puhrsch and R. Fergus, "Depth Map Prediction from a Single Image using a Multi-Scale Deep Network," in *NIPS*, 2014.

[9] Z. Hao, Y. Li, S. You and F. Lu, "Detail Preserving Depth Estimation from a Single Image Using Attention Guided Networks," in *International Conference on 3D Vision (3DV)*, 2018.

[10] P.-H. Huang, K. Matzen, J. Kopf, N. Ahuja and J.-B. Huang, "DeepMVS: Learning Multi-view Stereopsis," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[11] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy and T. Brox, "DeMoN: Depth and Motion Network for Learning Monocular Stereo," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[12] A. R. Zamir, A. Sax, W. Shen, L. Guibas, J. Malik and S. Savarese, "Taskonomy: Disentangling Task Transfer Learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[13] H. Zhou, B. Ummenhofer and T. Brox, "DeepTAM: Deep Tracking and Mapping," in *European Conference on Computer Vision*, 2018.

[14] C. Godard, O. M. Aodha and G. J. Brostow, "Unsupervised Monocular Depth Estimation with Left-Right Consistency," in *CVPR*, 2017.

[15] M. Menze and A. Geiger, "Object Scene Flow for Autonomous Vehicles," in *Conference on Computer Vision and Pattern Recognition*, 2015.

[16] X. Cheng, P. Wang and R. Yang, "Learning Depth with Convolutional Spatial Propagation Network," 2018.

[17] F. Zhang, V. Prisacariu, R. Yang and P. Torr, "GA-Net: Guided Aggregation Net for End-to-end Stereo Matching," in *CVPR*, 2019.

[18] X. Du, M. El-Khamy and J. Lee, "Deep Atrous Multiscale Stereo Disparity Estimation Networks," 2019.

[19] H. Fu, M. Gong, C. Wang and Batman, "Deep Ordinal Regression Network for Monocular Depth Estimation," in *CVPR*, 2018.

[20] R. Diaz and A. Marathe, "Soft Labels for Ordinal Regression," in *CVPR*, 2019.

[21] H. Ren, M. El-Khamy and J. Lee, "Deep Robust Single Image Depth Estimation Neural Network Using Scene Understanding," in *CVPRW*, 2019.

[22] opensource.com, "What is open source?," [Online]. Available: https://opensource.com/resources/what-open-source. [Accessed 24 Mayo 2019].

[23] GitHub Help, "Licensing a repository," [Online]. Available: https://help.github.com/en/articles/licensing-a-repository. [Accessed 24 May 2019].

[24] Open Source Guides, "The Legal Side of Open Source," [Online]. Available: https://opensource.guide/legal/#which-open-source-license-is-appropriate-for-my-project. [Accessed 24 May 2019].

[25] J.-R. Chang, "PSMNet," GitHub repository, 2018. [Online]. Available: https://github.com/JiaRenChang/PSMNet.

[26] I. Alhashim and P. Wonka, "DenseDepth," GitHub repository, 2018. [Online]. Available: https://github.com/ialhashim/DenseDepth.

[27] C. Godard, O. Mac Aodha and G. Brostow, "monodepth," GitHub repository, 2018. [Online]. Available: https://github.com/mrharicot/monodepth/.

[28] S. Park, J. Kim, R. Mizouni and U. Lee, "Motives and Concerns of Dashcam Video Sharing," in *CHI*, San Jose, California, 2016.

[29] España, "Real Decreto 1428/2003, de 21 de noviembre, por el que se aprueba el Reglamento General de Circulación para la aplicación y desarrollo del texto articulado de la Ley sobre tráfico, circulación de vehículos a motor y seguridad vial," *Boletín Oficial del Estado,* no. 306.

[30] España, "Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales.," *BOE,* no. 294, pp. 119788 - 119857, 2018.

[31] España, "Resolución de 30 de diciembre de 2016, de la Dirección General de Empleo, por la que se registra y publica el Convenio colectivo del sector de empresas de ingeniería y oficinas de estudios técnicos.," *BOE,* no. 15, pp. 4356-4382, 2017.

[32] España, "Ley 37/1992, de 28 de diciembre, del Impuesto sobre el Valor Añadido," no. 312, 2018.

[33] J. Zou and L. Schiebinger, "AI can be sexist and racist - it's time to make it fair," *Nature,* 2018.

[34] The Institute for Ethical Machine Learning, "The Responsible Machine Learning Principles," 2019. [Online]. Available: https://ethical.institute/principles.html. [Accessed 5 May 2019].

[35] M. Fahle, "Wozu zwei Augen?," *Naturwissenschaften,* vol. 74, pp. 383-385, 1987.

[36] I. P. Howard and B. J. Rogers, Binocular Vision and Stereopsis, New York: Oxford University Press, 1995.

[37] R. Descartes, Opera Philosophica, 1692.

[38] A. Fusiello, R. Vito and E. Trucco, "Efficient Stereo with Mutiple Windowin," in *Conference on Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, 1997.

[39] B. Musleh, A. de la Escalera and J. Armingol, "Detección de obstáculos y espacios transitables en entornos urbanos para sistemas de ayuda a la conducción basados en algoritmos de visión estéreo implementados en GPU," *Revista Iberoamericana de Automática e Informática Industrial,* no. 9, pp. 462-473, 2012.

[40] G. Bradski and A. Kaehler, Learning OpenCV, Sebastopol: O'Reilly, 2008.

[41] E. Kiperwasser, O. David and N. S. Netanyahu, "A Hybrid Genetic Approach for Stereo Matching," 2013.

[42] R. Washbourne, "What Is Stereo Matching?," DEVPY, 2015. [Online]. Available: https://www.devpy.me/what-is-stereo-matching/. [Accessed 6 June 2019].

[43] R. Alkemade, "Depth perception for augmented reality using parallel Mean Shift segmentation," Radboud University Nijmegen, 2011.

[44] R. Affendi Hamzah and H. Ibrahim, "Literature Survey on Stereo Vision Disparity Map Algorithms," *Hindawi Publishing Corporation Journal of Sensors,* vol. 2016, p. 23, 2016.

[45] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *International Journal of Computer Vision,* vol. 47, no. 1-3, pp. 7-42, 2002.

[46] J.-R. Chang and Y.-S. Chen, "Pyramid Stereo Matching Network," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pp. 5410-5418, 2018.

[47] A. Saxena, J. Schulte and A. Y. N. Ng, "Depth Estimation using Monocular and Stereo Cues," *IJCAI,* vol. 7, pp. 2197-2203, 2007.

[48] C. Holzmann and M. Hochgatterer, "Measuring Distance with Mobile Phones Using Single-Camera Stereo Vision," in *32nd International Conference on Distributed Computing Systems Workshops*, Macau, China, 2012.

[49] I. Alhashim and P. Wonka, "High Quality Monocular Depth Estimation via Transfer Learning," *arXiv e-prints,* vol. abs/1812.11941, 2019.

[50] J. West, "A Theoretical Foundation for Inductive Transfer".

[51] "ImageNet," [Online]. Available: http://www.image-net.org/. [Accessed 15 May 2019].

[52] L. Perez and J. Wang, "The Effectiveness of Data Augmentation in Image Classification," in *CVPR*, 2017.

[53] F. a. P. S. a. S. E. a. B. K. Luan, "Deep Photo Style Transfer," *arXiv preprint arXiv:1703.07511,* 2017.

[54] N. Silberman, D. Hoiem, P. Kohli and R. Fergus, "Indoor Segmentation and Support Inference from RGBD Images," in *ECCV*, 2012.

[55] A. A. P. B. E. B. Martín Abadi, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015.

[56] A. a. G. S. a. C. S. a. C. G. a. Y. E. a. D. Z. a. L. Z. a. D. A. a. A. L. a. L. A. Paske, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.

[57] Qualcomm Developer Network, "Qualcomm Neural Processing SDK for AI," Qualcomm, 2019. [Online]. Available: https://developer.qualcomm.com/software/qualcomm-neural-processing-sdk. [Accessed 8 June 2019].

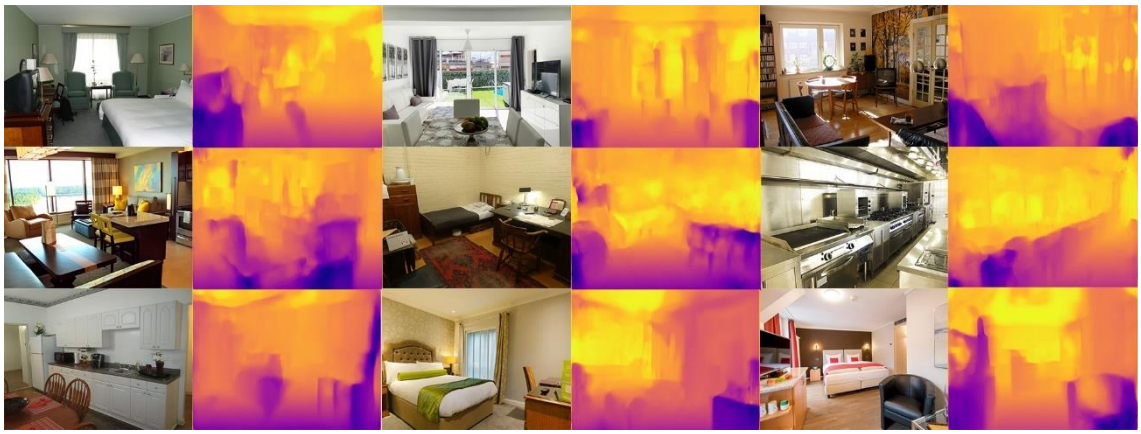
[58] C. Godard, O. Mac Adoha, M. Firman and G. Brostow, "Digging Into Self-Supervised Monocular Depth Estimation," 2019.

Glossary

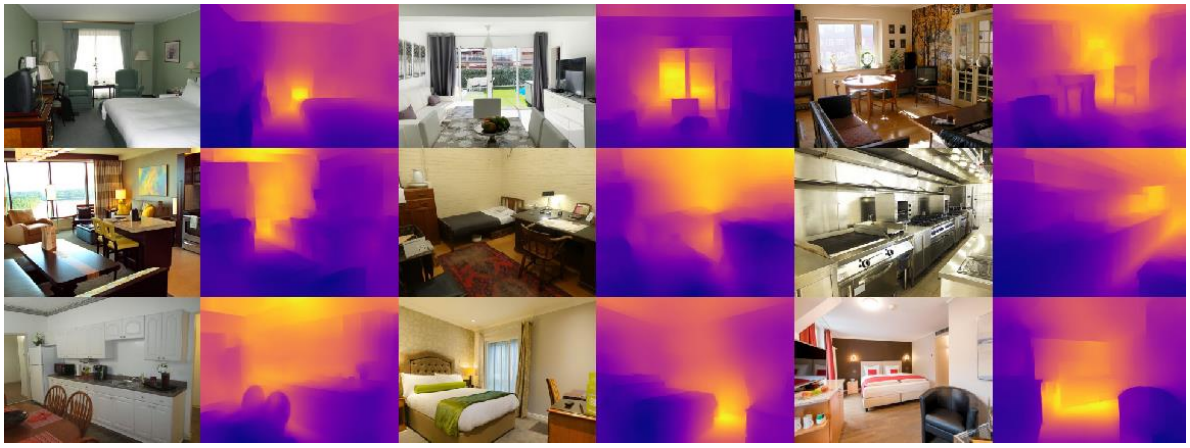
MLP	*Multi-layer Perceptron*
LIDAR	*Laser Imaging Detection and Ranging*
GPS	*Global Positioning System*
VAT	*Value Added Tax*
RGB	*Red, Green and Blue (Color channels)*
CNN	*Convolutional Neural Network*
SPP	*Spatial Pyramid Pooling*
CPU	*Central Processing Unit*
GPU	*Graphics Processing Unit*
VRAM	*Video Random Access Memory*
RAM	*Random Access Memory*

Appendix

Annex A. Monodepth (Indoor w/ KITTI & Roads w/ Eigen)



Annex B. DenseDepth (Indoor w/ KITTI & Roads w/ NYU)



Final annex

ORIGINALITY DECLARATION

Hereby I, Daniel Sarmiento Rocha, certify that the bachelor thesis titled “Estimation of Depth Maps from Monocular Images using Deep Neural Networks” is totally original mine, that has not been presented in any other university as a bachelor thesis and that all sources that have been used have been properly cited and appear in the references.

Colmenarejo, 17 June 2019

Signature: